EXHIBIT 4

IN THE UNITED STATES DISTRICT COURT
FOR THE NORTHERN DISTRICT OF CALIFORNIA
SAN FRANCISCO DIVISION

ORACLE AMERICA, INC.,	)	
	)	
Plaintiff,	)	
	)	
v.	)	Civ. A. No. 10-03561 WHA
	)	
GOOGLE INC.,	)	*(Jury)*
	)	
Defendant.	)	

EXPERT REPORT OF PROFESSOR DOUGLAS C. SCHMIDT, Ph.D.

January 8, 2016

Highly Confidential – Attorneys' Eyes Only

Table of Contents

a) Android will not function if the copied lines of declaring code are deleted from the source files of a single class of the copied 37 Java API packages at issue

95. As a first test, I took all of the copied declaring code out of the class java.text.annotation while leaving all other code intact. I then attempted to build the Android platform and it failed. When the build was attempted without each of the copied declarations, the build process returned an error and aborted prematurely every time. The complete error log summarizing the output of this build attempt can be found in Appendix M - Error Log from Android Build Test No. 3A: Removal of Source File Declaring Codes from java.text.annotation.

b) Android will not function if the copied lines of declaring code are deleted from the source files of the copied 37 Java API packages at issue

96. Next, I performed a final test by removing the copied declaring codes from all of the copied 37 Java API packages at issue. Once again, the output of the build for this test did not contain the necessary system image files. The findings from this test demonstrates that Android fails to build successfully when the copied declaring code lines are specifically removed, while leaving all other non-copied lines of code intact. These failures to build illustrate that Android is inoperable in the absence of any one of the copied declaring codes from inside the copied Java APIs. The complete error log summarizing the output of this build attempt can be found in Appendix N - Error Log from Android Build Test No. 3B: Removal of all copied declaring codes from all 37 APIs.

IX. THE ANDROID PLATFORM VIOLATES THE FUNDAMENTAL "WRITE ONCE, RUN ANYWHERE" PARADIGM OF THE JAVA PLATFORM

A. The Android Platform is incompatible based on the Java Compatibility Kit

97. I understand from the declaration of Mark Reinhold, Chief Architect of the Java Platform Group at Oracle, that Google's implementation of the 37 Java APIs in Android has failed to pass the official compatibility tests which are based on the Java Specifications.

98. The Java Compatibility Kit (JCK) is Java's official test suite for determining compatibility. It is used by Oracle to confirm compatibility of alleged implementations of Java with the adopted

specifications of the JCP.[36] I understand that the Java SE 6 platform passes all of the tests of the Java SE 6 Specification JCK, and that Java SE 7 platform passes all of the tests of the Java SE 7 Specification JCK. I also understand that failure of any one JCK test means that the alleged implementation is not compatible with the corresponding Java SE Specification.

99. A key test within the JCK is the Signature Test, which uses static analysis to determine whether or not an alleged implementation of 37 Java API contains the precisely correct number of classes, methods, interfaces, and fields. The test outputs a listing of missing and/or added class, constructor, field, method, nested class, super class, interface, and annotation elements, each of which constitutes an error. I understand from Dr. Reinhold's Declaration that any errors qualify an implementation as incompatible, i.e., any missing or added element renders the implementation incompatible. In particular, any missing element indicates the alleged implementation of Java represents a "subset" of the Java platform. Likewise, any added element indicates the alleged implementation of Java represents a "superset" of the Java platform.

100. It is my understanding that Dr. Reinhold used the JCK Signature Test to evaluate the compatibility of Android API Level 9 (Gingerbread), released in 2011, with Java SE 6, which was released near the end of 2006. Specifically, he ran the Java SE 6 JCK Signature Test on a version of the Java SE 6 runtime using Android Gingerbread's implementation of the 37 Java APIs.

101. It is my understanding that Dr. Reinhold also used the JCK Signature Test to evaluate the compatibility of Android API Level 21 (Lollipop), released in 2014, with Java SE 7, which was released near the end of 2011. Specifically, he ran the Java SE 7 JCK Signature Test on a version of the Java SE 6 runtime using Android Gingerbread's implementation of the 37 Java APIs.

102. Dr. Reinhold found that the Android Gingerbread's implementation of the 37 Java APIs in the Java SE 6 runtime failed to pass the Java SE 6 TJCK, generating a total of 409 errors for missing or added signatures. The results of this analysis are presented in Table 2 below, organized by type of error and package name.

[36] *See* https://jcp.org/en/resources/tdk, (last visited January 7, 2016), for a page on the Java Community Process (JCP) website describing how Specification Leads for JSRs obtain access to JCK testing tools; *See* http://openjdk.java.net/groups/conformance/docs/JCK6bUsersGuide/JCK6b_Users_Guide.pdf, (last visited January 7, 2016), for a copy of the user's guide for version 6b of the Java Compatibility Kit.

Table 2: Summary of error types for the 37 copied API packages for "addition" and "removal" errors from the Java SE 6 TCK against Android Gingerbread

	Error Type						
	Class		Method		Other		
Package Name	**Added**	**Missing**	**Added**	**Missing**	**Added**	**Missing**	**Total**
java.awt.font	0	1	0	5	0	0	**6**
java.beans	1	1	2	2	0	0	**6**
java.io	0	0	6	1	2	0	**9**
java.lang	2	2	37	12	7	4	**64**
java.lang.ref	1	0	1	0	2	0	**4**
java.lang.reflect	5	2	8	0	10	0	**25**
java.net	5	2	32	5	18	0	**62**
java.nio	2	0	10	0	2	0	**14**
java.nio.channels	0	5	0	28	0	1	**34**
java.nio.channels.spi	0	0	0	1	0	0	**1**
java.nio.charset	2	0	1	0	0	0	**3**
java.security	0	0	2	2	0	0	**4**
java.security.cert	0	2	3	1	0	1	**7**
java.security.spec	0	0	2	0	0	0	**2**
java.sql	0	0	0	14	44	0	**58**
java.text	0	0	5	2	1	0	**8**
java.util	5	3	21	22	3	3	**57**
java.util.jar	1	0	0	0	1	0	**2**
java.util.logging	0	0	1	1	0	0	**2**
java.util.prefs	1	0	1	0	0	0	**2**
java.util.regex	1	0	0	1	0	1	**3**
java.util.zip	4	0	2	0	0	5	**11**
javax.net.ssl	2	0	7	10	2	0	**21**
javax.sql	0	0	0	4	0	0	**4**
Total	**32**	**18**	**141**	**111**	**92**	**15**	**409**

103. Likewise, Dr. Reinhold found that the Android Lollipop's implementation of the 37 Java APIs in the Java SE 7 runtime failed to pass the Java SE 7 TCK, generating a total of 303 errors for missing or added signatures. The results of this analysis are presented in Table 3 below, organized by type of error and package name.

Table 3: Summary of error types for the 37 copied API packages for "addition" and "removal" errors from the Java SE 7 TCK against Android Lollipop

	Error Type						
	Class		Method		Other		
Package Name	**Added**	**Missing**	**Added**	**Missing**	**Added**	**Missing**	**Total**
java.awt.font	0	19	0	5	0	0	**24**
java.beans	1	35	1	1	0	0	**38**
java.io	0	0	0	1	0	0	**1**
java.lang	0	7	0	12	0	4	**23**
java.net	0	6	0	5	0	0	**11**
java.nio.channels	0	23	0	28	0	1	**52**
java.nio.channels.spi	0	1	0	1	0	0	**2**
java.security	0	3	2	2	0	0	**7**
java.security.cert	0	8	0	2	0	1	**11**
java.sql	0	1	0	14	20	0	**35**
java.text	0	0	2	2	0	0	**4**
java.util	5	3	9	22	1	3	**43**
java.util.logging	0	0	1	1	0	0	**2**
java.util.regex	0	0	0	1	0	1	**2**
java.util.zip	0	0	0	0	0	5	**5**
javax.crypto.spec	0	0	0	0	1	0	**1**
javax.net.ssl	0	2	0	10	0	0	**12**
javax.security.auth	0	3	0	0	0	0	**3**
javax.security.auth.callback	0	6	0	0	0	0	**6**
javax.security.auth.login	0	14	0	0	0	0	**14**
javax.security.auth.x500	0	1	0	0	0	0	**1**
javax.sql	0	2	0	4	0	0	**6**
Total	**6**	**134**	**15**	**111**	**22**	**15**	**303**

104. Table 4 below illustrates that Gingerbread and Lollipop's failures of the JCK Signature Test arise from their use of subsets of classes compared to Java SE 6 and 7, respectively. Since subsetting involves the absence of required classes, it will always result in failure of the JCK Signature Test.

Table 4: Summary of manual evaluation of "missing" method errors from the results of the Java SE 7 TCK test against the Android Lollipop runtime implementation

Class Name	Missing TCK Signatures from Lollipop Test	Java SE 7 Signature & Line #
java.util.Calendar	method public boolean java.util.Calendar.isWeekDateSupported()	Calendar.java #2224: *public boolean isWeekDateSupported()*
java.security.Permission	method public abstract boolean java.security.Permission.equals(java.lang.Object)	Security.java #135: *public abstract boolean equals(Object obj)*
java.net.URLConnection	method public long java.net.URLConnection.getContentLengthLong()	URLConnection.java #510: *public long getContentLengthLong()*
java.awt.font.NumericShaper	method public final java.util.Set<java.awt.font.NumericShaper$Range> java.awt.font.NumericShaper.getRangeSet()	NumericShaper.java #1216: *public Set<Range> getRangeSet()*
java.lang.ClassLoader	method protected static boolean java.lang.ClassLoader.registerAsParallelCapable()	ClassLoader.java #1243: *protected static boolean registerAsParallelCapable()*
java.io.File	method public java.nio.file.Path java.io.File.toPath()	File.java #2187: *public Path toPath()*
java.text.DecimalFormatSymbols	method public final static java.text.DecimalFormatSymbols java.text.DecimalFormatSymbols.getInstance()	DecimalFormatSymbols.java #127: *public static final DecimalFormatSymbols getInstance()*
java.util.regex.Matcher	method public final java.lang.String java.util.regex.Matcher.group(java.lang.String)	Matcher.java #520: *public String group(String name)*
java.util.logging.Logger	method public final static java.util.logging.Logger java.util.logging.Logger.getGlobal()	Logger.java #215: *public static final Logger getGlobal()*
java.sql.Connection	method public abstract int java.sql.Connection.getNetworkTimeout() throws java.sql.SQLException	Connection.java #1485: *int getNetworkTimeout() throws SQLException*

105. Likewise, Table 5 below illustrates that Gingerbread and Lollipop's failures of the JCK Signature Test also stem from their use of supersets of classes compared to Java SE 6 and 7, respectively. Since supersetting involves the addition of superfluous classes, it will always result in failure of the JCK Signature Test.

Table 5: Summary of manual verification of "added" errors from the results of the Java SE 7 TCK test against the Android Lollipop runtime implementation

Class Name	Added TCK Signatures from Lollipop Test	Android Lollipop Signature & Line #
java.beans.PropertyChangeListenerProxy	method public java.util.EventListener java.util.EventListenerProxy.getListener()	PropertyChangeListenerProxy.java #56: *PropertyChangeListener listener = (PropertyChangeListener) getListener()*
java.util.concurrent.ForkJoinPool	method public boolean java.util.concurrent.ForkJoinPool.awaitQuiescence(long,java.util.concurrent.TimeUnit)	ForkJoinPool.java #3023: *public boolean awaitQuiescence(long timeout, TimeUnit unit)*
java.util.concurrent.TimeUnit	method public final long java.util.concurrent.TimeUnit.convert(long,java.util.concurrent.TimeUnit)	TimeUnit.java #166: *public long convert(long sourceDuration, TimeUnit sourceUnit)*
javax.xml.validation.SchemaFactory	method public static javax.xml.validation.SchemaFactory javax.xml.validation.SchemaFactory.newInstance(java.lang.String)	SchemaFactory.java #180: *public static SchemaFactory newInstance(String schemaLanguage)*
java.util.ResourceBundle	method public static java.util.ResourceBundle java.util.ResourceBundle.getBundle(java.lang.String)	ResourceBundle.java #134: *public static ResourceBundle getBundle(String bundleName) throws MissingResourceException*
java.util.logging.Logger	method public static java.util.logging.Logger java.util.logging.Logger.getGlobal()	Logger.java #392: *public static Logger getGlobal()*
java.util.ResourceBundle$Control	method public static java.util.ResourceBundle$Control java.util.ResourceBundle$Control.getControl(java.util.List<java.lang.String>)	ResourceBundle.java #763: *public static Control getControl(List<String> formats)*

106. In conclusion, the Android platform's implementation of the 37 Java APIs fails the JCK Signature Test because it represents both a subset and superset of the Java SE API Specification. This failure establishes that Google's Android platform implementations deviate from the compatibility standards imposed by Java's testing suites.

B. Android's Runtime Environment and Virtual Machine are incompatible

107. A runtime environment broadly refers to the set of tools and resources that are used to execute applications that are created for a software platform. The Android and Java SE platforms

XI. ATTESTATIONS

123. I declare under penalty of perjury under the laws of the United States of America that the foregoing is true and correct. Executed on this 8th day of January, 2016, in San Francisco, California.

Douglas C Schmidt